Survey of Semantic Extensions to UDDI: Implications for Sensor Services

J. Caleb Goodwin, David J. Russomanno, and Joseph Qualls
Herff College of Engineering
Department of Electrical and Computer Engineering
The University of Memphis
Memphis, Tennessee 38152
{jgoodwin, drussmnn, jbqualls}@memphis.edu

Abstract*–The ability for software agents to discover, query, and task ubiquitous sensors requires machine-interpretable service descriptions, such as those proposed by the Semantic Web effort. Descriptions that support deep semantics will enable on-the-fly utilization of sensors for applications that might not have been anticipated on initial deployment. Semantic Web service discovery and dynamic composition requires formal semantic descriptions of inputs, outputs, preconditions, and effects of services. Universal Description Discovery and Integration (UDDI) provides a registry for publication and discovery of Web services, but it lacks the semantics needed for discovery and interoperation as envisioned by the Semantic Web community due to UDDI's syntax-based search. This paper surveys representative approaches for incorporating semantic capabilities within the existing UDDI infrastructure and then proposes an architecture for sensor services within an ontology-based network-centric environment.*

Keywords: Semantic Web, Sensors, Services, Ontology

1. INTRODUCTION

Network-centric environments require on-the-fly discovery and composition of Web services to satisfy an agent's high-level goals. The current standard specified by the Organization for the Advancement of Structured Information Standards (OASIS) for service discovery within a Service-Oriented Architecture (SOA) is Universal Description, Discovery, and Integration (UDDI) [1]. Despite wide commercial acceptance, UDDI does not facilitate autonomous discovery and interoperation of disparate Web services.

UDDI's search capability lacks support for semantic descriptions of Web services, which is integral in the dynamic discovery and composition process [2-9]. UDDI's search capability is syntax-based and relies solely on XML, which enables syntactic, but not semantic query without the additional layers offered by Semantic Web infrastructure. Syntax-based matching lends itself to application-specific software development where reuse of Web services by other organizations is arduous. For instance, the syntactic description of a Web service's output could be syntactically different than the input of a candidate service for composition, but the two services may be semantically compatible. However, an agent would not be able to discern the similarity without semantic descriptions and a possible composition would be missed.

Semantic search allows for the generalization and specialization of user queries through the understanding of the relationships between ontological objects and services. For instance, semantic understanding of sensor performance capabilities, operational capabilities, and physical properties may aide in the discovery of sensor services through generalization or specialization of the initial query. The investment in UDDI makes the augmentation of semantic capabilities within its existing infrastructure of great interest to those in Homeland Security, Department of Defense, or others involved in government and private sector applications.

Dynamic utilization of sensor Web services and fusion of data obtained from distributed heterogeneous sensor deployments may facilitate the discovery of knowledge, which is unattainable from unitary sensor percepts [10-13]. Classification of the sensed objects may require discovery and orchestration of available sensor services for fusion of high- and low-level percepts. The dynamic discovery and interoperability of sensor services requires a semantic sensor service broker.

Although there are a variety of approaches to capturing semantics in SOAs, including [14], this paper surveys representative approaches within the UDDI/SOA context. The remainder of the paper is organized as follows. Section 2 provides an overview of Web services, UDDI, and Semantic Web service discovery. Section 3 evaluates the existing approaches to augment UDDI with semantic search capabilities. Section 4 proposes an architecture for a sensor service broker within an ontology-based sensor network environment, which is being implemented in a laboratory environment.

2. WEB SERVICES, UDDI, AND SEMANTICS

Web services are distributed functional units accessible via a network environment that are modular and self-describing. A Web service is identified by a universal resource identifier (URI) and contains an interface description describing the capabilities of the module, communication protocol, and port locations.

Figure 1 depicts the generic model for an SOA using UDDI which provides standards for description, discovery, connection, and communication between Web services. The architecture consists of the following: i) service provider that registers descriptions of services; ii) broker that maintains a registry of services and providers; and iii) service requestors who search the registry for available services [15].

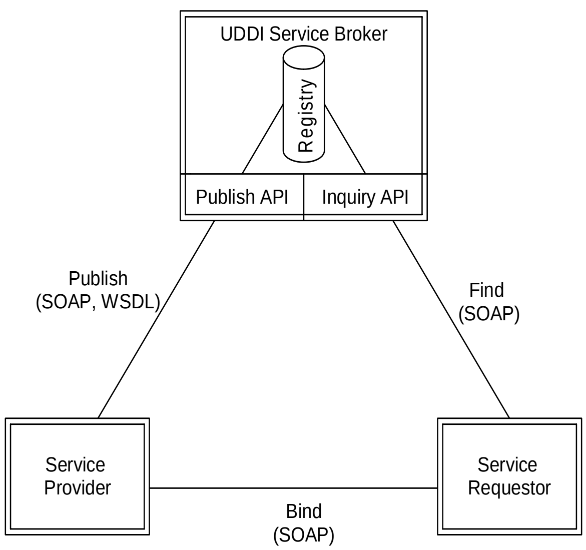

Figure 1. Generic model for Web services [15].

The following description of UDDI follows closely from [15]. UDDI is an on-line registry where service providers can register Web services providing descriptions that can later be discovered by service requestors. UDDI provides standard APIs to query and publish Web services using Simple Object Access Protocol (SOAP) for transmitting messages. Web Service Description Language (WSDL) provides XML descriptions for inputs, outputs, bindings, etc. The WSDL service descriptions are mapped to the UDDI data structures where they are later searched when fulfilling service requests. A series of predefined SOAP messages are used to search for and register services in the UDDI registry.

UDDI's core XML data structures consist of business service, service provider, and service binding objects. Each of the data objects contain text fields and technical models (tModels) for additional metadata inclusion. TModels allow external links to resources outside of the UDDI framework and allow for extensions to the overall system.

Due to the underlying reliance solely on XML, UDDI is limited to keyword searches and lacks inference capabilities even if the tModel for a given object links to a taxonomy or ontology providing further semantic information. The shortcomings of UDDI's searching capabilities are two-fold: i) searching is syntax based; and ii) there is no guarantee that the returned service is correct even if it matches the keyword search due to the lack of semantics. For instance, keyword searching for an *Infrared_Camera* will not return the service entry for a *Midwave_Camera* even though midwave cameras are a specialization of infrared cameras.

Adding semantics to UDDI would strengthen its overall search capabilities. For instance, in a military application there may be a need to search as follows: "Find an infrared camera in location X with FOV 22x45 degrees and lens of 13mm." The current UDDI infrastructure cannot accommodate such a sophisticated query much less find generalizations or specializations of any portion of the query. A query of high complexity requires generalization and specialization capabilities and access to instance specific metadata for subsequent reasoning. Semantically similar objects, while not satisfying the query directly, may aid in satisfying an agent's overall goal.

3. CURRENT ARCHITECTURE OVERVIEW

A classification of representative approaches for augmenting UDDI with semantic capabilities is shown in Figure 2. The major classifications are as follows: modify UDDI API, internal mapping of semantic schema to UDDI data structures, external matchmaker extensions, and referencing external semantic schema.

The modification of the UDDI API allows for the extension of the existing search capabilities and the semantic registration of services. Such modifications are often coupled with an internal semantic schema, external semantic schema, or semantic matchmakers for query enhancement.

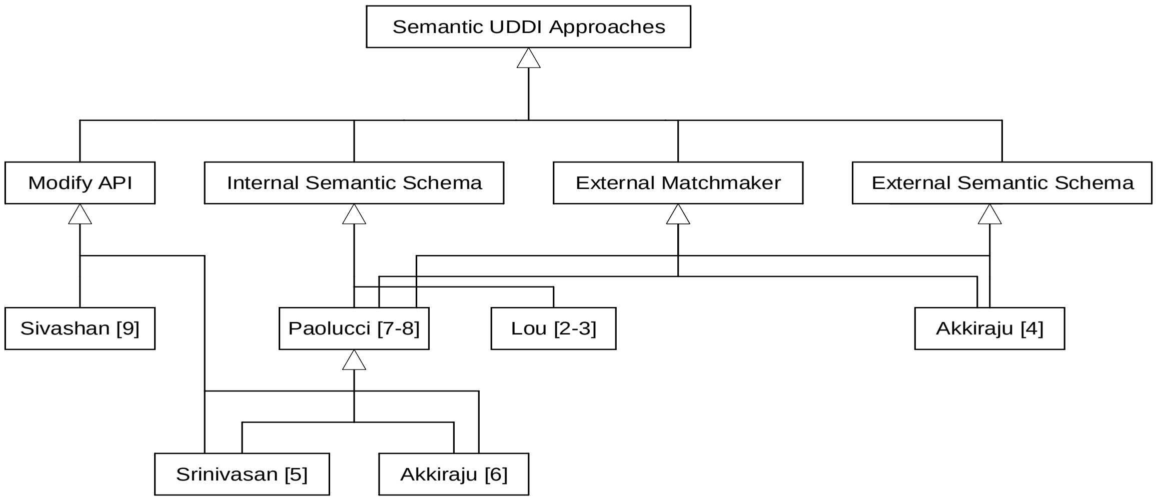

Figure 2. Classification of representative approaches for augmenting UDDI with semantic capabilities.

Loading semantic content into the UDDI registry involves using the tModels to map the semantic schema into the UDDI data store. Enhancing the API or augmentation of UDDI with a matchmaker can then traverse the tModel's for inference, generalization, specialization, or to load additional semantic information. The external matchmaker resides outside of the UDDI data store and is invoked when a request is received. The semantic matcher provides query expansion and generalization/specialization to locate semantically similar services. Storing external schema involves the use of the tModel data structure. The provider registers the service and then provides a URI so that the semantic matcher can access the external semantic schema for subsequent processing.

Paolucci et al. [7-8] specify the mapping of Semantic Web service descriptions into tModels, external semantic schema, and augmenting UDDI with a matchmaker. The description of the service provider is mapped directly into UDDI using a series of 15 tModels such as: *providedBy*, *inputTModel*, *outputTModel*, etc. The matchmaker module resides on a different port location than that of UDDI and is loosely coupled with the existing infrastructure. The add-on module intercepts the UDDI requests, interprets them, and then facilitates communication with the UDDI registry. An external ontology is referenced by the DAML-S matching engine for generalization of the input/output parameters associated with registered Web services.

An advertisement message is mapped and registered in UDDI providing an ID for the service. The ID of the registered service and the capability of the service are sent to the matchmaker where the information is used in conjunction with the stored ontology to process queries. A service request is passed to the matchmaker where the ID of the service is extracted, if a satisfactory service is available. UDDI is then queried and the information pertaining to the service is returned to the requestor.

Srinivasan et al. [5] extend the architecture proposed by Paolucci and the UDDI API. Srinivasan presents a more tightly coupled architecture whereby the add-on semantic matching module resides on the same server and uses the UDDI registry's publish and inquiry ports for processing requests. Srinivasan adds a capability port to the UDDI API that allows for capability generalization by the matchmaker.

Akkiraju et al. [6] extend the architecture of Paolucci by adding an inquiry API specification to UDDI. The inquiry API allows a service requestor to specify the request in semantic markup language describing the input, output, and categorical semantics. This approach allows for generalization of the request through the use of semantic schema contained by the matchmaker.

Sivashanmugam et al. [9] present an approach similar to Paolucci for the mapping of semantic schema to UDDI. TModels are used to link from the concept to the domain ontology and provide an interface to construct queries that use semantic annotation. The user enters Web service requirements as templates constructed using ontological concepts. The algorithm matches Web services based on functionality, similarity rating, precondition, and post-condition effects. The functionality search selects a subset of services and then uses input/output semantics to prune the search space.

The architecture developed by Akkiraju et al. [4] uses tModel's to point to additional external RDF semantic information, such as ontologies or instance specific semantic schema. The semantic matcher is registered in UDDI as a service and can access the external semantic schema for inference. The matcher is invoked dynamically based upon its specialized domain allowing for selection of the best matcher given the query criteria. The architecture allows for registry use, with or without semantic matching, depending on the requirements of the service requestor. The matchmaker architecture is viewed as a service so the inference capabilities are constricted only by the implementation of the matchmaker on the semantic schema.

The approach presented by Luo et al. [2-3] bulk loads ontologies into the UDDI data store via extensive use of tModels. An ontology aware matchmaker used by the client can then make use of the ontology stored in UDDI as well as service instances that reference the stored ontology. The OWL-S service description for a new service maps to the UDDI data model objects. The tModels in the data objects reference the ontology loaded into UDDI. The realization of the system is analogous to developing an object-oriented conceptual model, but losing semantics when implementing the conceptual design in a relational database. For instance, OWL language constructs such as subPropertyOf, Class, and subClassOf are retained, but the semantics entailed from minCardinality, maxCardinality, and FunctionalProperty are lost. Furthermore, the reliance upon the naïve query capabilities of UDDI will further limit the inference capabilities of an agent.

4. SEMANTIC SENSOR BROKER

4.1 Architecture Overview

Distributed sensor deployments can have a myriad of communication methods requiring common protocol for interoperation. SOA provides the standard communication protocol as well as methodology to discover, invoke, and publish sensor Web services. As shown in Figure 3, the sensor service provider includes the base stations that serve as a connectivity point between the sensor service requestor and the physical sensor(s). The sensor service provider registers its respective services with a broker, aggregates and stores data from its respective sensor deployments, and processes queries either by retrieving archived data or tasking the deployed sensors in its network. The sensor service broker in our architecture references the OntoSensor sensor ontology [11, 13].

OntoSensor was developed using Protégé 2000 [16] for describing sensors on the Semantic Web for autonomous processing by intelligent agents. OntoSensor is an ontology of sensor types and describes sensor attributes and operational parameters. OntoSensor includes knowledge models for the data acquisition boards, sensing elements, and processor/radio units contained in the Crossbow 2006 catalog [17], as well as preliminary definitions of a variety of imaging sensors.

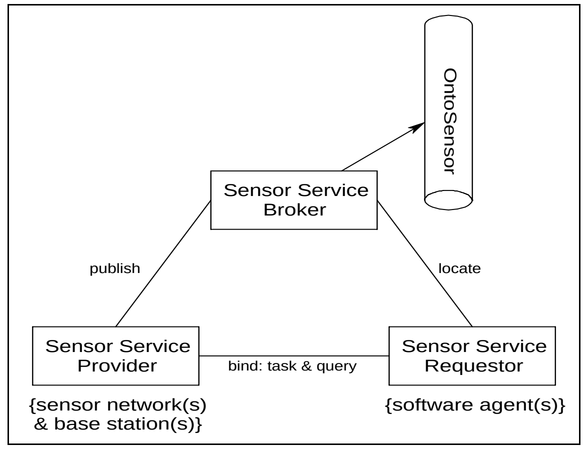

Figure 3. Generic model for sensor services.

The architecture for the prototype sensor service broker is shown in Figure 4. The framework leverages the architecture proposed by Akkiraju et al. [4] and allows for maximum inference capabilities by the agent, attainment of instance specific semantic schema, and multiple matchmakers. The adopted approach avoids precluding various inference capabilities, such as functional properties [2-3] and provides semantic capabilities beyond input/out generalization [7-8]. Furthermore, the architecture allows for domain-specific matchmakers and attainment of external semantic schema required for sensor service discovery not present in some proposed architectures [2-3, 5-9].

The sensor service interface facilitates communication between the agent, matchmaker, and UDDI. The sensor service interface allows for extension of the UDDI API while keeping the standard API intact. If the user desires semantic matching, UDDI is searched to discover the external ontologies of the registered services. The sensor service interface then searches UDDI to find matchmakers that provide matchmaking services within the domain of a given ontology. If a matchmaker is discovered, the sensor service interface invokes the matchmaker which performs semantic matching on the query in conjunction with the registered services in UDDI.

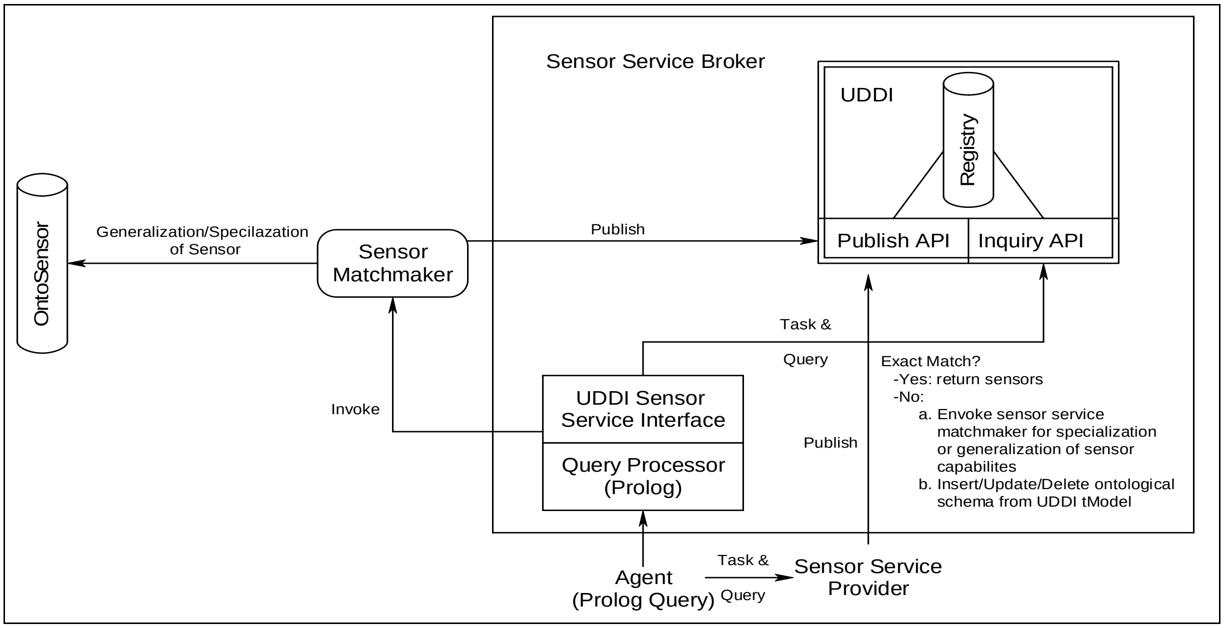

Figure 4. Sensor semantic service description and match approach for prototype.

A semantic sensor matchmaker is proposed that will utilize OntoSensor to aid in discovery of available sensors and sensor Web services. The semantic sensor matchmaker can be viewed as an instance of a matchmaker Web service. Regarding matchmakers as instances allows for future addition of sensor matchmakers that can specialize within a domain of sensor types or composition of sensor services. For instance, sensor Web services can be composed of specific algorithms that take raw sensor data as input and provide a classification based on the raw data as output. This framework allows for specialized matchmakers that can work in conjunction to resolve locating the appropriate sensors while others can specialize in orchestration of classification applications to achieve a high-level goal.

It is a goal that OntoSensor will ultimately contain the majority of the conceptual model required for semantic search of sensors. However, specific sensor instances contain properties that the matchmaker will have to load at query time as required. For instance, the geo-location, orientation, and lens type will vary among the various instances of thermal infrared imaging cameras. An external reference methodology is adopted to provide external links to instance specific external semantic data [4]. The DescribedUsing tModel allows for instances to reference additional semantic schema by including a pointer to the location of the external description. The external description is then loaded by the matchmaker when needed.

A robust matchmaker is required for sensor service discovery and is beyond the scope of input/output matching or simple capability matching. Queries, such as supported applications, geo-location, resolution, day/night operation, and instance specific properties such as lens type, etc. must be supported. For example, the semantic sensor matchmaker receives the following request: "Infrared_Camera location=xxx." The matchmaker first searches UDDI to see if there are any instances of Infrared_Camera. In this scenario, UDDI does not contain any entries. The matchmaker then finds all subclasses of Infrared_Camera and searches UDDI again. FLIR_ThermaCam is found in the registry which is a subclass of Infrared_Camera. The matchmaker then obtains the external link to the instance specific semantic schema and determines that the camera is in location "xxx." The bindings of the services are then returned to the service requestor.

4.2 UDDI as Cache

The UDDI registry shown in Figure 4 can be viewed as a cache. If the service requester's query is satisfied directly by the service specifications in UDDI then the sensors' bindings are returned; otherwise the sensor matchmaker is invoked to find generalizations or specializations that are semantically similar to the initial query. Ontological knowledge pertinent to the semantically similar sensor is updated/inserted into the UDDI tModels. The functionality is similar to a conventional cache whereby subsequent queries can be satisfied using the registry without the aid of the matchmaker. The sequence diagram in Figure 5 depicts the use of UDDI as a cache to satisfy a query for a sensor having the requested capability.

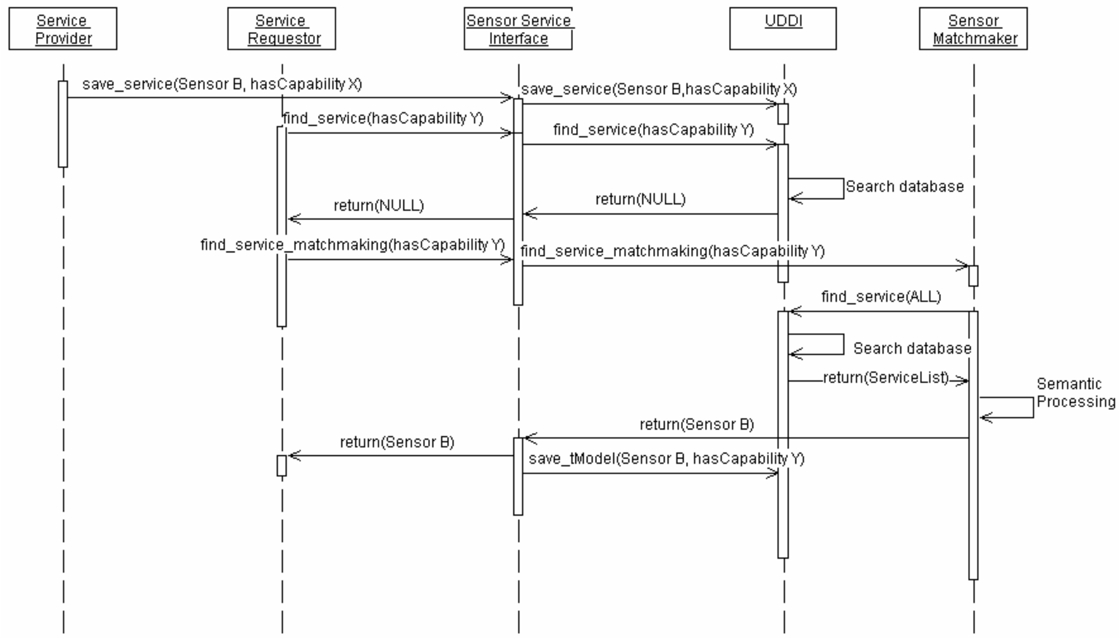

Figure 5. Sequence diagram for semantic search.

The approach does not seek to seamlessly integrate or bulk load OntoSensor into UDDI, but it provides a means to enhance the syntactic matching initially used for queries. For instance, in Figure 5 the request for a sensor service with capability Y initially fails, but it is satisfied using semantic matching in conjunction with the semantic knowledge about sensors in Figure 6, which is an excerpt from OntoSensor. The matchmaker uses the semantic knowledge to deduce that Sensor X is a subclass of Sensor Y; therefore, it also has capability Y. A subsequent analogous query succeeds without matchmaking due to the explicit assertion of these deduced facts in the UDDI database.

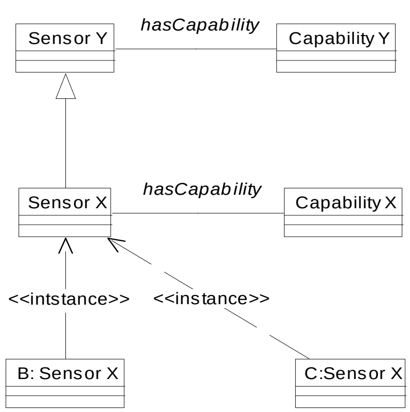

Figure 6. Excerpt of ontological sensor knowledge.

Augmenting UDDI with semantic metadata after semantic matchmaking can be thought of as a heuristic approach to query satisfaction. Consider the following events in which steps 1-8 correspond with Figure 5:

1. A service provider registers sensor B with capability X
2. A service requestor queries for a sensor with capability Y
3. UDDI is searched and no hits are found for the request
4. A service requestor requests matchmaking for a sensor with capability Y
5. The matchmaker discovers that sensor B is an instance of Sensor X which has capability Y
6. UDDI is searched for sensor B and it is found
7. The bindings for sensor B are returned to the requestor
8. UDDI data for sensor B is augmented so that it is explicit that it has capability Y
9. A service provider registers sensor C with capability X
10. A service requestor queries for a sensor with capability Y
11. Sensor B is found in UDDI and returned to the service requestor (sensor C is not found)

In step 11, sensor C is not found in UDDI by the service requestor's initial syntax-based query (step 10). On initial inspection, this may appear to be an oversight. However, the returned services can be evaluated by the requesting agent to determine if they fulfill the needs of the agent. If the returned services are not satisfactory, the agent will request again, but for a deeper probe, that is, semantic matching. For instance, a subsequent deeper probe, as shown by operation N in Figure 7, would discover that sensor C has capability Y. The process is analogous to using a set of compiled heuristics first (that is, the explicit facts in the UDDI repository) then, falling back on deeper knowledge (the sensor ontology) if the heuristics do not provide an acceptable solution.

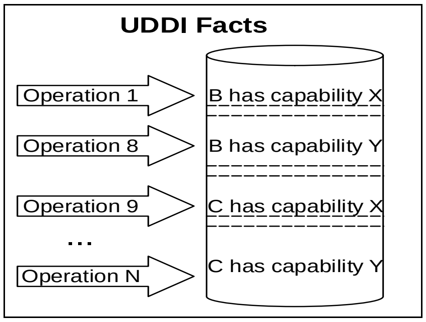

Figure 7. UDDI database.

5. CONCLUSION

Representative architectures for augmenting UDDI with semantic search capabilities have been discussed in the context of sensor service discovery. An architecture for semantic sensor matchmaker was then proposed. The architecture can be viewed as an instantiation of the Akkiraju et al. [4] framework, which allows for multiple, domain-specific matchmakers, retrieval of instance-specific semantic knowledge, and maximum matchmaking inference. Future work seeks to further implement the proposed architecture for sensor discovery in an ontology-based network-centric environment.

ACKNOWLEDGEMENT

Funding was provided in part by agreement W911NF-05-2-0019 between the U. of Memphis and the U.S. Army's Research Lab (ARL). This paper does not necessarily represent the position of the U.S. Government.

REFERENCES

[1] L. Clement, A. Hately, C. Riegen, and T. Rogers, "UDDI Version 3.0.2," OASIS, Tech. Rep., 2004. Available at: http://uddi.org/pubs/uddi v3.htm

[2] J. Luo, B. Montrose, and M. Kang, "Adding Semantic Support to Existing UDDI Infrastructure," Report No. NRL/MR/5540--05-8918, Naval Research Laboratory, Code 5542, 2005.

[3] J. Luo, B. Montrose, and M. Kang, "An Approach for Semantic Query Processing with UDDI," *1st International Workshop on Agents, Web Services, and Ontology Merging*, Agai Napa, Cyprus, 2005.

[4] R. Akkiraju, J. Colgrave, K. Verma, P. Akkiraju, and R. Goodwin, "Dynamic Discoverery and Binding of Web Services to Abstract Web Process Flows," *International Journal of Web Services Research*, to appear.

[5] N. Srinivasan, M. Paolucci, and K. Sycara, "Adding OWL-S to UDDI, Implementation and Throughput," *First International Workshop on Semantic Web Services and Web Process Composition*, San Diego, CA, 2004.

[6] R. Akkiraju, R. Goodwin, P. Akkiraju, and S. Roeder, "Method for Semantically Enhancing the Service Discovery Capabilites of UDDI," *Workshop on Information Integration on the Web, IJCAI*, Acapulco, Mexico, 2003.

[7] M. Paolucci, T. Kawamurea, T. Payne, and K. Sycara, "Importing the Semantic Web in UDDI," *Proceedings of Web Services, E-Business, and Semantic Web Workshop*, pp. 225-236, 2002.

[8] M. Paolucci, T. Kawamura, T. Payne, and K. Sycara, "Semantic Matching of Web Services Capabilites," *International Semantic Web Conference*, Sardinia, Italy, 2002.

[9] K. Sivashanmugam, K. Verma, A. Sheth, and J. Miller, "Adding Semantics to Web Service Standards," *Proceedings of the 1st International Conference on Web Services*, Las Vegas, NV, pp. 395-401, 2003.

[10] G. Jiang, W. Chung, and G. Cybenko, "Semantic Agent Technologies for Tactical Sensor Networks," *2003 SPIE Conference on AeroSense*, Orlando, FL, pp. 21-25, 2003.

[11] D.J. Russomanno, C. Kothari, and O. Thomas, "Building a Sensor Ontology: A Practical Approach Leveraging ISO and OGC Models," *The 2005 International Conference on Artificial Intelligence*, Las Vegas, NV, pp. 637-643, 2005.

[12] S. Avancha, C. Patel, and A. Joshi, "Ontology-driven Adaptive Sensor Networks," *First Annual International Conference on Mobile and Ubiquitous Systems, Networking and Services*, pp. 194-202, 2004.

[13] C. Goodwin and D.J. Russomanno, "An Ontology-Based Sensor Network Prototype Environment," *Fifth International Conference on Information Processing in Sensor Networks*, Nashville, TN, 2006.

[14] J. Liu and F. Zhao, "Towards Semantic Services for Sensor-Rich Information Systems," *2nd IEEE/CreateNet International Workshop on Broadband Advanced Sensor Networks*, Boston, MA, 2005.

[15] M. Singh and M. Huhns, *Service-Oriented Computing*, West Sussex, UK: John Wiley & Sons, 2005.

[16] N. Noy, M. Sintek, S. Decker, M. Crubezy, R. Fergerson, and M. Musen, "Creating Semantic Web Contents with Protégé-2000," *IEEE Intelligent Systems*, vol. 16, pp. 60-71, 2002.

[17] Anon., *Wireless Sensor Networks: Product Reference Guide*, CrossBow Technology Inc., 2006.